What is claimed is:

1. A method of allocating registers when compiling source code, said method comprising steps of:

translating source code to intermediate code;

identifying an operand from said intermediate code to store in a real register; and

selecting a class of real registers operable to store said operand.

2. The method of claim 1, further comprising steps of:

selecting at least one subclass of said selected class of real registers, wherein said at least one subclass includes a register to store said operand.

3. The method of claim 1, wherein said selected class includes one of a callee-saved class and a caller-saved class.

4. The method of claim 2, wherein said step of selecting at least one subclass further comprises steps of:

selecting a first set of subclasses within said selected class;

determining whether a register included in said first set of subclasses is available to store said operand; and

in response to said register being available, storing said operand in said register.

5. The method of claim 4, wherein said first set of subclasses includes at least one of non-used-in-current-operation, non-busy, non-live and non-used subclasses.

6. The method of claim 4, wherein said step of selecting at least one subclass further comprises steps of:

selecting a second set of subclasses within said selected class in response to said register not being available in said first set of subclasses;

determining whether a register included in said second set of subclasses is available to store said operand; and in response to said register in said second set of subclasses being available, storing said operand in said register in said second set of subclasses.

7. The method of claim 6, wherein said second set of subclasses includes at least one of non-used-in-current-operation, non-busy, non-live and used subclasses.

8. The method of claim 6, wherein said step of selecting at least one subclass further comprises steps of:

selecting a third set of subclasses within said selected class in response to a register in said second set of subclasses not being available;

determining whether a register included in said third set of subclasses is available to store said operand; and

in response to said register in said third set of subclasses being available, storing said operand in said register in said third set of subclasses.

9. The method of claim 8, wherein said third set of subclasses includes at least one of non-used-in-current-operation, live and non-busy subclasses.

10. The method of claim 8, wherein said step of selecting at least one subclass further comprises steps of:

selecting a fourth set of subclasses within said selected class in response to a register in said third set of subclasses not being available;

determining whether a register included in said fourth set of subclasses is available to store said operand; and

in response to said register in said fourth set of subclasses being available, storing said operand in said register in said fourth set of subclasses.

11. The method of claim 10, wherein said fourth set of subclasses includes at least one of non-used in current operation and busy subclasses.

12. The method of claim 11, further comprising spilling a register in at least one of said busy and said live subclasses prior to storing said operand in said register in at least one of said busy and said live subclasses.

13. The method of claim 11, further comprising storing said operand in a class other than selected class in response to a register in said fourth set of subclasses not being available.

14. The method of claim 11, further comprising marking said register as used-in-current-operation in response to storing said operand in said register.

15. The method of claim 11, further comprising marking said register storing said operand as live and not-used-in-current-operation in response to translating an instruction of said source code.

16. The method of claim 1, further comprising steps of:

selecting another class of registers in response to said selected class of registers not including a not used in current operation register; and

storing said operand in a register in said selected other class.

17. The method of claim 3, wherein said step of selecting a class further comprises steps of:

selecting said callee-saved class in response to said operand including at least one of local variables, stack items and parameters input by a user; and

selecting said caller-saved class in response to said operand including a temporary computation.

18. A method of compiling source code comprising steps of:

generating intermediate code from a portion of source code;

allocating a plurality of real registers to store a plurality of operands from said intermediate code while generating the intermediate code; and generating machine-readable code from said intermediate code using said plurality of real registers.

19. The method of claim 18, further comprising a plurality of types of operands and said step of allocating further comprises steps of:

determining a type of operand for at least one of said plurality of operands;

storing said at least one operand in memory in response to said operand being a particular type of operand; and

allocating a real register for said operand.

20. The method of claim 19, wherein said particular type of operand includes a local variable.

21. The method of claim 19, wherein said step of allocating further comprises steps of:

selecting a class of registers depending on said type of operand; and

allocating a real register from said selected class of registers depending on said type of operand.

22. The method of claim 21, wherein said step of selecting a class further comprises steps of:

selecting a first class of registers in response to said operand being at least one of a local variable, a stack item and a parameter input by a user; and

selecting a second class of registers in response to said operand being a temporary computation.

23. The method of claim 21, wherein said step of selecting allocating further comprises selecting at least one subclass of registers in said selected class.

24. The method of claim 23, wherein said at least one selected subclass includes at least one of live registers, non-live registers, busy registers, non-busy registers, used registers, non-used registers, and non-used in current operation registers.

25. A compiler configured to compile source code into machine-readable code, said compiler comprising:

a register allocation stage configured to generate intermediate code from said source code and configured to allocate a plurality of real registers to a plurality of operands from said intermediate code;

an optimization stage configured to optimize said intermediate code; and

a final code stage configured to generate said machine-readable code from said intermediate code using said plurality real registers.

26. The compiler of claim 25, wherein said register allocation stage is configured to determine a type of operand for at least one of said plurality of operands, and store said at least one operand in memory in response to said operand being a particular type of operand, and allocate a real register for said operand.

27. The compiler of claim 26, wherein said particular type of operand includes a local variable.

28. The compiler of claim 25, wherein said register allocation stage is further configured to select a class of registers and allocate a real register from said selected class of registers for one of said plurality of operands, said one operand being of a particular type of operand.

29. The compiler of claim 28, wherein said register allocation stage is further configured to select a first class of registers in response to said operand being a type including at least one of a local variable, a stack item and a parameter input by a user; and

select a second class of registers in response to said operand being a temporary computation.

30. The compiler of claim 28, wherein said register allocation stage is further configured to select at least one subclass of registers in said selected class.

31. The compiler of claim 30, wherein said at least one selected subclass includes at least one of live registers, non-live registers, busy registers, non-busy registers, used registers, non-used registers, and non-used in current operation registers.